우리 프로젝트 : “드론은 수동(teleop)으로 이동하며 Gazebo에서 얻은 x,y 좌표를 GUI에 전달하고, GUI가 그 좌표를 터틀봇3의 Nav2에 전달하여 해당 지점으로 자율주행하게 하는” 과정을 구현하는 개략적인 방법. 드론 좌표 획득 → GUI 전달 → 터틀봇 Nav2 Goal 로 이어지는 ROS2 노드 설계와 좌표계 관련 유의사항에 초점을 맞추겠습니다.

<문제점1>
지피티야 내가 world에서 맵을 다시 다시딴거를 위의 zip파일안에 src안에 description안에 world에 넣었는데 ros2 launch sjtu_drone_bringup sjtu_drone_bringup.launch.py이거를 터미널에서 실행했을때 새로 수정한 맵이 반영이 안되. 어디가 문제인거지?
<해결방안1
sjtu_drone_bringup.launch.py) 내에서 사용하고 있는 world 파일의 경로가 실제 수정한 파일의 경로와 일치하는지 확인해보니까 문제가 해결됨.

<문제점2>
지피티야 근데 warehouse.world의 배경안에 model.sdf와 model.config의 것도 넣어야하는데 어떻게 해야해? 참고로 다음의 것들은 model.sdf와 model.config야(이것들은 역시 warehouse.world랑 같은 위치에 있어)=위에 mines폴더는 model.sdf와 model.config 이 들어있는데 위의 사진을 참고해서 어떻게 내가 해야 저 박스들까지 같이 gazebo안에 띄울수 있을까?
<해결방안2>
warehouse.world 최종 코드 (4개 박스 포함)를 만들고
이후에 환경 변수 설정 (GAZEBO_MODEL_PATH)을 해줌(과정은 아래 두줄과 같음)
폴더를 그대로 쓰기 위해서는 Gazebo가 mines 폴더를 인식해야 합니다.
다음 명령어로 현재 터미널에서 warehouse_mines_v0.0 폴더를 모델 경로로 추가하세요.

bash
Copy
cd /home/seungrok/ros222_ws/src/sjtu_drone-ros2/sjtu_drone_description/worlds
export
GAZEBO_MODEL_PATH=$PWD/warehouse_mines_v0.0:$GAZEBO_MODEL_PATH
이렇게 하면 model://mines를 Gazebo가 찾을 때, warehouse_mines_v0.0/mines 폴더를 확인하게 됩니다.
근데 위의 방법이 ros2 launch…으로 실행할 때는 동일한 쉘 환경이 유지되지 않아 model://mines를 찾지 못해서
--->mines 폴더를 ~/.gazebo/models에 복사하해서
현재 mines 폴더를 ~/.gazebo/models 폴더로 옮기면, Gazebo가 기본 경로에서 모델을 찾음
mkdir -p ~/.gazebo/models
cp -r /home/seungrok/ros222_ws/src/sjtu_drone-ros2/sjtu_drone_description/
worlds/warehouse_mines_v0.0_w_color/mines ~/.gazebo/models
결론적으로 이제 환경 변수 설정 없이도 model://mines를 찾을 수 있으므로, ros2 launch sjtu_drone_bringup sjtu_drone_bringup.launch.py 실행 시 박스가 잘 뜨게됨

<문제점3>
지피티야 위에는 gazebo에서 drone을 띄우는 zip파일이야.
일단 여기에다가 turtlebot3을 drone하고 같이 띄우려고 하는데 내가 무엇을 더 하면 좋을까?
<해결방안3>
드론 패키지 내 launch 폴더(예: ~/ros2_ws/src/sjtu_drone_bringup/launch)에 새 파일을 만듭니다. 파일 이름은 combined.launch.py을 만듬
-> 드론 패키지 & 터틀봇3 패키지를 같은 워크스페이스 안에 넣었다.(그리고 당연히 setup.py도 설정을 다시했지요(빌드하고 나서)

<문제점4>
turtlebot3(waffle모델)에 manipulation이 달리지 않았음
<해결방안4>
#robot_state_publisher가 manipulation URDF(turtlebot3_manipulation.urdf.xacro)를 읽어 "robot_description" 토픽에 퍼블리시하게 함.즉 robot_state_publisehr 런치 파일을 수정함
#spawn_turtlebot3.launch.py를 수정함(터틀봇3를 Gazebo에 스폰하는 파일에서는 기존에 -file 옵션으로 SDF 파일을 직접 로드하던 부분을 -topic 옵션으로 바꿔줌)

<문제점5> turtlebot3에 manipulation이 뜨긴하는데 갑자기 turtlebot3가 날아다님ㅋㅋ(드론이 날아다녀야하는데)
<해결방안5> 둘이 쓰는 토픽이 겹쳐서 생긴 문제점이였음.(즉 두 로봇이 같은 토픽을 사용해서 충돌이 발생한거지)---> 각 시스템에 고유한 네임스페이스를 부여했음.
드론 쪽 launch 파일에서는 네임스페이스를 drone으로 지정했음
그러면 드론 관련 토픽은 /drone/cmd_vel을 지정
터틀봇3 쪽 launch 파일에서는 네임스페이스를 turtlebot3으로 지정(터틀봇3 관련 토픽은 /turtlebot3/cmd_vel이 됨

<문제점6> 드론의 초기위치 설정
<해결방안> 드론과 터틀봇의 spawn launch 파일(예, spawn_drone.launch, spawn_turtlebot3.launch)을 찾아 수정하면 Gazebo 상의 초기 위치를 원하는 좌표(-1.013570, 3.022503, -0.000001)로 설정했음

<문제점7> 드론의 초기설정을 바꾸니까 드론의 teleop이 되지 않았음
<해결방안> teleop.py, teleop_joystick.py 의 두파일을 보니 'cmd_vel', 'takeoff', 'land' 토픽에 명령을 퍼블리시하고 있는데, 드론 URDF의 gazebo 플러그인에서는 네임스페이스를 /simple_drone으로 사용합니다.
즉, 드론 플러그인이 /simple_drone/cmd_vel, /simple_drone/takeoff, /simple_drone/land 등을 구독하도록 설정되어 있으므로, teleop 노드도 해당 절대 토픽으로 퍼블리시하도록 변경하니까 해결됨

<문제점8> 근데 갑자기 build가 안되었음

<해결방안> CMakeList.txt를 수정 : src/worlds 폴더를 설치하려고 할 때 경로가 달라 빌드 에러가 발생했으나, CMakeLists.txt의 install() 구문을 수정해 문제를 해결

<문제점9>
지피티야 위에 zip파일과 test.py는 드론이 gazebo상에서 world를 정찰하고 test.py에 드론의 bottom카메라가 gui상에 연결되는것을 보여주기위한 test.py야. test.py에서는 drone이 정찰할때 gazebo상에서 world를 정찰하는 bottom카메라가 보이고 제거시작을 누르면 turtlebot3가 drone의 위치에 가는거야.
근데 우리의 문제점은 turtlebot3가 drone의 위치로 가는게 아니라 그냥 test.py안에 지정된 좌표로 움직이는것이 우리의 문제점이야.
나는 이 문제점을 해결하고 싶어. 이러한 문제점을 해결하기 위해서 너한테 ros2222_ws안에 src의 tree를 줄테니까 어디를 수정해야 우리의 문제점이 해결될수 있는지 해결방법을 자세하게 알려줘.
turtlebot3가 drone의 위치에만 가면 우리의 프로젝트는 완성이 되는거거든.
<해결방안9> test.py를 살펴보니 그냥 직선으로만 쭉가게…되어있었음.(고정 위치점이 박혀있었네…)

<문제점10> 아니 manipulation이 땅아래로 박혀있었음
즉 로봇팔이 다른 각도로 떨어져있었음
<해결방안10>
turlebot3_manipulation_bringup/gazebo.launch와 combined.launch.py의 두부분에서의 두 launch파일 사이에 로봇팔 초기화 부분의 차이가 있었음.(초기 joint세팅 부분에서의 문제점)
“turtlebot3_manipulation_bringup/gazebo.launch.py”의 내용을 combined.launch.py에 병합해버림

<문제점11>터틀봇이 직선으로는 안가긴하는데 걍 막감…(우리가 원하는 위치로 안감 gazebo상에서 좌표로 가지않음)
<해결방안11> 현재 test.py는 /drone_position 토픽을 구독하여 드론의 좌표(self.drone_x, self.drone_y)를 업데이트하도록 되어 있습니다. 그런데 실제 드론의 위치가 이 토픽으로 올바르게 퍼블리시되지 않으면, test.py에서는 초기값(예: 0.0 혹은 고정된 값)이 그대로 사용됩니다.-> 해결 방법

드론 제어 노드 수정:
sjtu_drone_control 패키지 내의 드론 위치 관련 노드(예: drone_pose_subscriber.py 또는 drone_position_control.py)에서 드론의 pose 데이터를 받아 geometry_msgs.msg.Point 메시지로 변환하여 /drone_position 토픽에 퍼블리시하도록 수정

불필요한 수동 Twist 명령 제거
문제점
startRemoval 함수 내에서는 goal_pose를 퍼블리시한 후,

로봇을 정지시키고
2초 대기 후에 수동으로 전진 명령(Twist)을 퍼블리시하고 있습니다.
이 수동 Twist 명령은 내비게이션 스택의 경로 계획 및 제어를 방해할 수 있습니다. 결과적으로 turtlebot3가 미리 정해진 (고정된) 방향으로만 움직일 수 있습니다.

해결 방법

내비게이션 위임:
목표 위치(goal_pose)를 퍼블리시한 후에는 내비게이션 스택이 경로 계획 및 이동을 담당하도록 하며, 별도로 전진 명령을 주지 않습니다.

<문제점12> 로봇이 근데 이제는 움직이지를 않음
<해결방안12>
1. 내비게이션 스택을 사용하지 않는 경우
현상
로봇이 /goal_pose를 구독하지 않으므로, PoseStamped를 퍼블리시해도 로봇이 움직이지 않습니다.

해결책

직접 Twist 명령을 퍼블리시

test.py에서 “제거 시작” 버튼을 누르면, 해당 좌표로 향하도록 경로 계산 + Twist 명령을 직접 구현해야 합니다.
예: move_cmd.linear.x, move_cmd.angular.z 등을 계산해서 로봇을 이동시키는 로직을 작성.
기존 내비게이션 없이 직접 이동

예를 들어, 드론 위치가 (x, y)에 있다고 하면, 거기까지 자율주행 알고리즘(또는 간단한 P 제어 등)을 사용하여 Twist를 퍼블리시하는 방식을 구현해야 합니다.

<문제점13> 근데 갑자기 tf_transformations내부에서 np.float를 사용하지 않는다고 해서
<해결방안>13gui코드에서 np.float를 builtin float로 재정의함
#import numpy as np
#np.float = float